PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 :
G06F 13/00

A1

(11) International Publication Number: WO 00/19324

(43) International Publication Date: 6 April 2000 (06.04.00)

(21) International Application Number: PCT/US99/22331

(22) International Filing Date: 28 September 1999 (28.09.99)

(30) Priority Data:
60/102,019 28 September 1998 (28.09.98) US

(71) Applicant: ARGUS SYSTEMS GROUP, INC. [US/US]; 1809 Woodfield Drive, Savoy, IL 61874 (US).

(72) Inventors: McNABB, Paul, A.; 2116 Bristol Road, Champaign, IL 61821 (US). SLAVIN, Pavel, S.; 4009 Turnberry Drive, Champaign, Il 61821 (US). HANSON, Chad, J.; 2320 E. University #8, Urbana, IL 61802 (US). SANDONE, Randall, J.; 1779 County Road 1550 N., Urbana, IL 61802 (US).

(74) Agent: BARKUME, Anthony, R.; Greenberg Traurig, Met Life Building, New York, NY 10166 (US).

(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

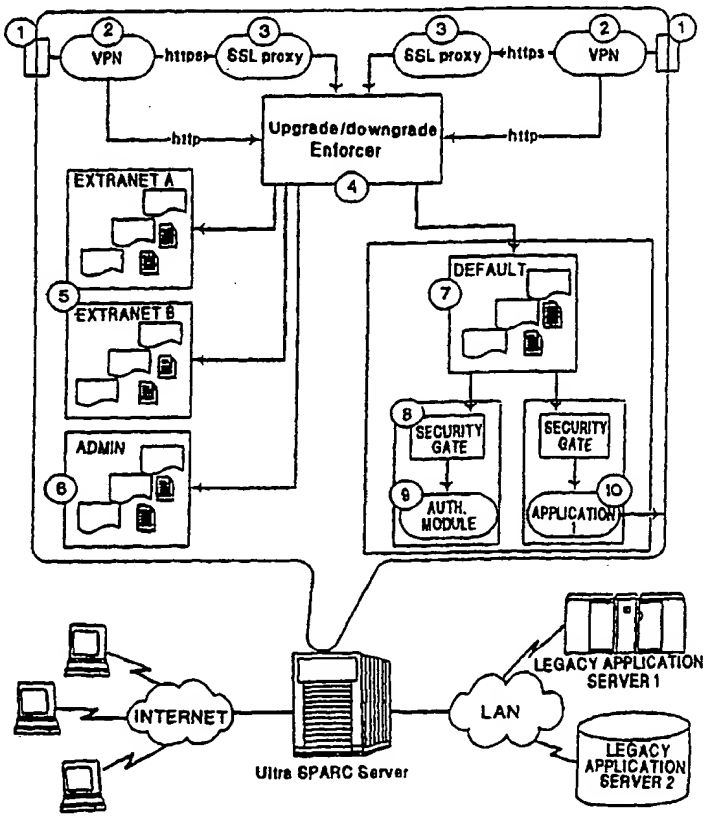
**Published**

*With international search report.*
*Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.*

(54) Title: TRUSTED COMPARTMENTALIZED COMPUTER OPERATING SYSTEM

(57) Abstract

A system and method for providing a trusted server which controls access to the execution of processes by applying file level extended sensitivity label attributes (202). The attributes are utilized to restrict execution of processes (250) that are requested by comparing the extended attributes (200) in addition to using standard file permission authorization. The system additionally may be used to provide controlled execution of commercially available software.



# TRUSTED COMPARTMENTALIZED COMPUTER OPERATING SYSTEM

## CROSS REFERENCE TO RELATED APPLICATIONS

This application is based on and claims filing priority of co-pending U.S. Provisional Application Serial Number 60/102,019, filed on September 28, 1998, which is incorporated by reference herein.

## TECHNICAL FIELD

This invention generally relates to computer systems security, and operating system design where the access, control, rights and privileges are assigned to the individual file members and not strictly to the user or process that accesses the computer. The system comprises operating system modifications to affect the access and control of processes executing on the server.

## BACKGROUND ART

The importance of a secure networking platform (such as one for the Internet) is underscored by the following example. In early November 1988, a self-replicating program was released upon the Internet, invading VAX and Sun-3 computers running versions of Berkeley Unix. This program exploited the resources of these computers to attack other computers connected to the Internet. Within hours, this program spread across the United States, infecting 6,000 of the 60,000 existing Internet hosts. At that time, the Internet was still used almost exclusively for exchanging mail among scientists. When organizations' Internet services were limited to static web pages, mail gateways, and the like, security measures were needed

critical services to the organization and connects private and public systems and data. For example, under this new business model, systems that once provided only publicly available information to the Internet at large are now a potential doorway to confidential data such as bank account information or transaction records for computer hackers anywhere in the world.

On any computer system, certain system programs or utilities must be granted the ability to bypass the security constraints normally imposed by the system. For example, in order to create a backup of all files on the system disks, an administrator must be able to run a backup program that is able to read all files on the disk, even though the administrator would not normally be allowed such access. Other powerful programs must also be carefully controlled, such as the programs to shut down the system, create new users, and repair damaged file systems. On a standard Unix system, the operating system has been designed so that one user ID, called root or superuser, can bypass all security restrictions and limitations. Windows NT systems exhibit similar vulnerabilities with the 'System' and 'Administrator' accounts.

A utility that needs to use any restricted feature must therefore be run as root or administrator. This means for example, that the backup program can be exploited and used to shut down the system, and the shutdown program can be exploited to create new users, and the program to create new user accounts can be exploited to read all files on the system. Thus if any administration program has an

with malicious intent discover and exploit unknown holes in applications or operating system software.

These products generally operate on the misconception that an authorized and authenticated user is also a trustworthy user. Consider, for example, a malicious banking customer in possession of a valid account number and PIN. Traditional security measures recognize him as an authorized, legitimate user of the system. Once allowed access to the Internet server, this account holder could attack the server and use it as a bridgehead for entry into back-end databases and financial servers.

In studies conducted by several well-known computer industry analysts, security managers have indicated that they feel that the most significant threat to the integrity and security of their systems comes from malicious abuse and misuse by authorized persons inside the organization. These statistics demonstrate that effective security solutions must address the issue of protecting systems from insiders and others that are authorized to be using the systems as well as against determined attacks from trained and knowledgeable attackers.

Firewalls, by limiting access to host systems and services, provide a necessary line of perimeter defense against attack. Firewalls do not, however, adequately reduce the risk for applications that generate active content or implement transaction-oriented services. As the term implies, a firewall restricts overall access from a hostile environment (the Internet) to a friendly environment (the local company network). The new paradigm of transaction-based Internet services makes these "perimeter"

system into accepting their false identities, and the system and all its resources are rendered defenseless.

Intrusion detection is a tool for responding to attacks on a system. It relies on the system's ability to detect known patterns of activity associated with malicious intent. By definition, such a detection system is unable to deal with new exploits. For example, an attack that uses apparently innocuous packets to exploit previously unknown system bugs in the server will probably go unnoticed. Intrusion detection mechanisms are purely reactive; they do nothing to prevent the initial breach from occurring. Once a security hole has been exploited, application and operating system files are open to subversion, allowing an attacker to open other, undetected, security holes. Furthermore, attackers who successfully gain unfettered access to system audit trails can often delete system traces of their intrusion, effectively rendering the intrusion detection mechanism useless in the most severe cases of attack.

Most IS and corporate managers, already hard-pressed to maintain daily systems operations, face significant barriers to incorporating new technologies and adequate systems security. Managers seeking to upgrade security on their systems are thus often forced to rely on vendor claims of security performance. As new software emerges and inevitable upgrades to existing software pour in, IS professionals typically assume that the vendors have a vested interest in the security of their products. Given the potential implications of security system failure, it is critical that managers concentrate on security solutions

integration work typically associated with systems of this type.

It is an object of the present invention to provide a secure operating system for use on a firewall or information server where the access is strictly controlled and where the processing is restricted to permit only those actions required to respond to the request. It is another object of the invention to provide a server system where the administrative processes are controlled and executed for a local machine only, such that network users cannot access or modify the administrative functions of the system from outside the local network. It is an object of the present invention that the authorization of a user to request data from the system is compared to a role established for the user making the request. It is an object of the present invention that the request by a user may only initiate predefined processes where the authorization to perform a process is verified at each process step, and where the process does not inherit rights or pass rights to other subsequent processes. It is another object of the present invention that requests for the same item by different users may result in the users being routed to different locations where they are returned different results. It is another object of the present invention that file permissions are modified such that extended attributes are assigned to each file and executable process where these attributes are subsequently examined whenever a request is received by the system.

It is additionally desirable that a web server may be adapted to work with this operating system where a

comprising the steps of: receiving an incoming request for a data object; assigning a sensitivity label to an incoming request for a data object; reading extended attributes at a first storage destination associated with the data object; redirecting the incoming request to a second storage destination for the data object based on the combination of the sensitivity label and the extended attributes; executing an action associated with the redirected request.

A method is additionally provided for assigning control and access attributes to data objects for a commercial software product executing on a trusted server, where access to the commercial software product is restrictively managed by the trusted server, the trusted server performing the steps of: executing the commercial software product in a configuration mode; determining at least one process to be accessed by the commercial software product while in the configuration mode; receiving administrator input for a least one sensitivity level to be assigned to the data files and processes; determining extended attributes to be applied to the processes and data files of the commercial software product; applying the determined extended attributes for the received administrator sensitivity level to the processes and data components of the commercial software product; and storing the extended attributes for use in a non-configuration mode.

Following the configuration mode, a method is provided for executing the commercial software product on the trusted server of in a non-configuration mode further comprising the steps of: receiving a request related to the

a sensitivity label to an incoming request for a data object; determining a first destination associated with the incoming request for the data object; reading extended attributes at the first destination associated with the data object; and redirecting the incoming request to an alternate destination for the data object based on the combination of the sensitivity label of the incoming request and the extended attributes. The method further comprises generating the output requested which may ultimately comprise another process request that is handled in the same manner or sending output to the requestor.

A trusted server computing system is provided for permitting controlled execution of processes in response to a request comprises: storage means for storing a plurality of data objects with extended attributes in at least one data partition; processor means for receiving requests and executing processes in response to the user requests; assignment means for assigning a sensitivity level associated with the request for a data object; upgrade downgrade enforcer means for interpreting the request to determine a first destination to direct processor to retrieve the process and extended attributes, the upgrade downgrade enforcer compares the attributes to the assigned sensitivity level, and passes the favorably compared processes to the processor for execution; the processor identifies processes that require access to secured storage partitions and directs these processes along with the sensitivity level to a security gate means; the security gate means receives the sensitivity level and interprets processes requiring data from partitioned storage locations, and the security gate means retrieves the requested data if

Figure 16 is a data flow diagram of the processing steps of the present invention.

## BEST MODE FOR CARRYING OUT THE INVENTION

The present invention is a fully integrated software foundation for secure business processes. It incorporates a variety of security technologies into a seamless, secure system. Its components and modifications comprise: operating system security enhancements, network packet management modifications, upgrade/downgrade enforcer (UDE), security gate, trusted administration utilities, enhanced secure shell (enhanced SSH), authentication module, secure CGI module, network layer encryption (VPN), and usage of Secure Socket Layer encryption (SSL).

The following glossary is useful in conjunction with the description of the preferred embodiment herein:

Advanced Secure Networking (ASN): A security component associated with the network interface which assigns a label to each incoming packet to indicate which compartment or partition the packet belongs to.

Audit trail: A set of records that collectively provide documentary evidence of processing. The audit trail enables tracing of events forward from the original transactions to related records and reports, and backward from records and reports to their component source transactions.

Internet Protocol (IP): A standard specifying how devices connected over the Internet to uniquely identify themselves and communicate with one another.

ITSEC (Information Technology Security Evaluation Criteria): An internationally recognized set of standards for the evaluation, testing, and certification of information technology security products.

Least privilege: The principle that each subject in a system must be granted no more privileges (no higher clearance) than needed for the performance of authorized tasks. The application of this principle limits the damage that can result from accident, error, or unauthorized use.

Mandatory Access Control (MAC): A means of restricting access to objects based on the sensitivity (as represented by a label) of the information contained in the objects and the formal authorization (i.e., clearance) given to the particular request to access information of such sensitivity.

Multilevel secure: A class of system that permits access to various sets of information with different sensitivities by users with the correct specific security clearances and needs to know, but that prevents users from obtaining access to information for which they lack authorization.

Packet labeling: The process of attaching information associated with the permission level of the requesting process to an incoming or outgoing datagram.

Secure Socket Layer (SSL): A mechanism for providing session layer encryption for a web user.

directories, files, etc.); and 3) interprocess communication messages (including packets, shared memory, etc.). On a standard system, each of these has various security attributes, which are created, managed, and used by the OS itself. When a process attempts to access a file system object, the OS compares various attributes of the process with attributes of the object, and allows or denies access. When a process sends or receives communication messages, the OS verifies that the process is allowed to send and/or receive the message. When objects are created, such as when a file is created or when a message is generated, the OS is responsible for ensuring that the proper attributes are attached to the new object.

The trusted server system has extended this standard mechanism in two ways: by attaching additional security attributes to each of the OS components, and by extending the security checks to use the new attributes. A security attribute that has been added to all three types of components is the "sensitivity label" 202 or SL. Figure 2 shows the extended attributes applied to files while Figure 3 shows the attributes that are applied to the processes or packets that are processed on the system. Figure 4 shows the attribute types in a tabular form that are part of this structure. Unlike standard security mechanisms, the SL 202 is designed to be mandatory, i.e., not under the control of the user. This means that though a user may own a file, he would be unable to make it, or its contents or even a copy, accessible to a user that was not previously authorized to have such information. SLs can be related in several ways: 1) they can be equal, 2) one can be "greater" (dominates) than another, and 3) they can be "disjoint" (meaning neither is greater than the other). A simple analogy may help

file and write each onto a tape or disk, even though the administrator may not normally have access to all files. Under these circumstances, the process's "privilege" attributes are used to allow it to bypass specific security mechanisms. Privileges can be stored on a program's executable file so that when the program is run, the process running it will have the special capabilities it needs (these privileges are called "innate privileges" on the present system).

In addition to traditional authorizations and privileges, the trusted server system has added the concept of adding at least one set of privileges to a program file in addition to its innate privilege set, along with an additional list of authorizations, called the "privilege authorization set" 220. When a program is executed, the process will be given both the innate privileges and the "authorized privileges" if the user running the program has one or more of the privilege authorizations. This allows the administrator to set up programs so that users are not just allowed or denied access to the programs, but also so that some users will run the program with a greater ability to bypass certain security restrictions. Since this mechanism is applied to the copy of the program file stored on the disk, not to the program itself, this can be added to commercial, binary copies of software for which there is no access to the source code.

The trusted server of the trusted server system has added a minimum 254 and maximum clearance (SL) 256 for each process. This additional attribute is used to limit certain privileges. For other systems using a privilege mechanism and providing SLs, the privilege to ignore the SL

services such as HTTP, FTP, telnet, etc.) and/or protocol (e.g., TCP or UDP) along with the interface, source network/subnet, and source IP address. In addition, trusted server has allowed the administrator to have separate rules for packets leaving the system and packets coming into the system. These rules are used not only to provide default information for incoming, unlabeled packets, but also for determining if a packet should be allowed out or in on a specific network interface or to a specific host or network. The processes or components of this system may be implemented using software components, or may alternatively utilize microprocessors having embedded processes.

In the preferred embodiment of the Unix implementation, the trusted server system comprises a computer executing the kernel process of the present invention where in addition to the default inode structure information, a link is included to retrieve previously stored attribute label information related to the file. In another embodiment, the inode structure is modified to permit the direct inclusion of the label information such that it is accessible to the processing routines of the system of the present invention without looking at other information stored in a different portion (partition) of storage. The kernel is adapted to retrieve this label information whenever a process calls a file from storage. This label may enable the processor to retrieve the file stored at that reference location directly if the user's base authority and permissions permit the execution of the file and if the extended attributes in the system authorizes the user to access that level of information. The system comprises storage means to store the process and

be present due to a successful authentication attempt by the user authentication module.

A Security Gate 8 (see Figure 1) program is a special program that allows limited communication between two processes, or between a process and a network interface. Its configuration file specifies the source SL and port, along with the destination SL and port. In a typical configuration, the source and destination SLs are disjoint, meaning that the two ends cannot interact in any way. The Security Gate program is given an SL that is greater than both. It is also given an SL range that includes both endpoints, along with the privilege to override the SL security checks, but limited to only the SLs within its range.

The UDE (Upgrade/Downgrade Enforcer) program is more sophisticated than the Security Gate. It allows an incoming packet (which has been assigned security attributes as it arrived to the system) to be routed to another port, possibly with its SL being modified. The UDE uses a configuration file and a file of "rules" to determine what accesses are allowed. The UDE can pass HTTP traffic or other protocols. If HTTP traffic arrives, the UDE can examine the packet for 1) the presence of a "cookie" in the header, 2) the URL path specified, 3) destination port, and 4) the SL of the packet. Based on this information and on the rule database, the UDE can select a new SL for the packet, as well as a new port number and network address. Multiple copies of the UDE can be running on a single system. The UDE has the privileges it needs to be able to change the SL of traffic flowing through it.

in a secured operation mode and where the file storage structure and processes executed are modified to permit a highly secure method of managing content on the computer system.

The term data object is used to generically describe any content upon which a process is to be performed, and it may additionally be applied to the process itself such that parameters of an event to be executed are checked prior to the execution of that process step. This system may be implemented where each event to be performed is handled by one central event controller that comprises each of the aforementioned components (ASN, UDE, Security Gate) or alternatively each of the components may be separately executed such that they operate in a cooperative manner. The trusted operating system of the present invention supports SLs and uses them to determine if a user or process can access certain objects or resources. Since SLs cannot be modified by users or unprivileged processes, a system's security policy can be enforced without relying on user compliance. This access control using system-enforced labels is called mandatory access control (MAC).

Other restrictions implemented to secure the operating system of the present invention require the segmentation of the superuser (root) privilege, where the root account is not permitted to execute all of the processes associated with system administration. Instead this privilege is broken down into many smaller privileges. Thus the backup program may be able to read any file, but it cannot be exploited to shut down the system, modify files,

authorizations may be determined and applied prior to routing the communications to a preferred destination path or to a network service component. For incoming packets from the Internet for example, a label may determined at step 310 based on the host id requested at step 300 (see Figure 5), the protocol requested at step 304 and the port number at step 306. If no corresponding security label exists for that combination received, a check is performed to determine for a network interface whether a protocol match (at step 314) and port range (at step 316) exist. If a match is found at step 316 the corresponding security label is applied to the packet at step 322. This packet is then routed preliminarily to the UDE where based on the label previously applied, the packet is directed to the corresponding output destination that listeners are configured to listen to. The label may be modified at this time by the UDE to redirect the packet according to a more comprehensive set of rules.

Figure 8 shows a diagram of the processing steps for outgoing packets that uses similar verification steps to either drop the outgoing packet or to allow the packet to be sent if the host (step 400), protocol (step 402), port range (step 404), and SL (step 410) and network interface (step 414) are known to the system. Outgoing packets will have the label of the process or daemon that created them. An incoming or outgoing packet will be dropped if the SL of the packet is not valid for both the interface and the remote host. In addition, the SL can be inserted into the packet header so that trusted operating systems can share security information over the network.

according to content of the cookie stored on the user computer.

The Security Gate 8 (see Figure 1) program as previously mentioned is a special program that allows limited communication between two processes, or between a process and a network interface. Its configuration file specifies the source SL and port, along with the destination SL and port. For example, requests that are processed at the web server 500 that need access to other information in secured partitions are handled by the Security Gate 504 (see Figure 9) where a request 502 at an original SL may be received by the Security Gate 504 that may be operating at a higher SL level. Requests that are authorized may have their SLs modified from $SL_1$ to $SL_2$ in order to conform to the permitted SLs of the Back-end Database Server 510. As previously mentioned in a typical configuration, the source and destination SLs are disjoint, meaning that the two ends cannot interact in any way. The Security Gate 504 program is therefore given an SL that is greater than both. It is also given an SL range that includes both endpoints, along with the privilege to override the SL security checks, but limited to only the SLs within its range. In this manner the data from different partitions or computers of the system may be joined together as required for the purpose of generating data in response to the request. On the path back to the user the Security Gate performs the label translations required to return the replies to the requesting web server 500.

In an example of this system for a Unix operating system, a modified Init kernel is substituted for the

least one of the accesses listed in the access authorization set (280). (A separate authorization database lists which UIDs have which authorizations.)

If any of the prior verification steps failed, permission to execute the binary program would be denied.

In step 612, the trusted server compares the authorizations associated with the process UID (262) with the privilege authorization set (282). If the process UID (262) has one or more of the authorizations listed in the privilege authorization set (282), then the trusted system executes the binary with privileges listed in the file's innate privileges (286) and authorized privileges (284) at step 620. If the process UID (262) does not have any of the authorizations listed in the privilege authorization set (282), then the trusted system executes the binary with the privileges listed in the file's innate privileges (286), but not those in the privilege authorization set (282) at step 616.

The combination of UID authorizations in the authorization database and privilege sets of the file determine the privileges that are granted process B that results when process A executes a file. A particular process UID (262) for example, may be granted the MAKEIDB authorization in the authorization database. (The MAKEIDB (280) value may refer to a defined role that would permit processes with that authorization to perform certain actions.) If the process UID (262) had the MAKEIDB authorization (280), then when the process executed the file in Figure 4, the file would execute. If the process UID had

The authentication module 9 of the trusted server system can be configured to request a user to provide a user ID and a site-definable authentication response (such as a password, a biometric device, a smart card, or an access token check) prior to permitting access. Once the user has been authenticated, subsequent web requests can be identified by the UDE as part of an authenticated session, and communication to other restricted partitions can be allowed.

The data flow in the architecture of Figure 16 is illustrated here using three examples: an extranet environment, back-end/legacy application support, and use of the authentication component.

Referring to Figure 13, the present invention can be configured as an extranet server hosting two or more sensitive web sites that must be isolated from each other. The virtual separation of extranet web servers allows administrators to configure the system so that users accessing it with the same URL will get different web pages. This functionality is transparent to the user.

An example of this configuration is a company web server that can be accessed by employees, customers, shareholders, and others. The sensitivity of information varies for each category, and data protection becomes a critical issue. In this case the secure server is used to provide secure extranet functionality for three web servers (Extranet A, Extranet B, and Extranet C) that are running on the system. The components involved are VPN 20 via SKIP, the upgrade/downgrade enforcer 4, and the trusted server 22.

passed to the upgrade/downgrade enforcer (UDE) at step 140. The UDE 4 will parse the URL request and determine which web server should receive it. Then the UDE 4 changes the SL of the request to match that of the destination web server and transmits the request to the appropriate web server (at either step 142 or 144) if the permissions, access authorizations permit the change.

This approach defeats any attempt to directly connect to the extranet servers; access is granted only through the UDE 4. All communication to and from extranet servers is mediated by the UDE. Because the servers operate in isolated environments, any attempt to exploit bugs in the extranet server software cannot compromise any other application on the server.

This system then may be configured to provide secure connectivity between a web interface and a back-end application using the Security Gate. The web server serves the sole purpose of displaying information from a back-end application in a graphical format and is meant to be accessible by a public or a corporate network. For example, a bank may wish to let all of its customers use an Internet banking feature. Because of the threat that some of its customers may be malicious, sensitive data must be protected by separating the web server and the back-end application into separate virtual environments. This system provides a secure way to address these issues. With the security gate 8 component, there is no direct data flow between web server and back-end applications. All requests go through the security gate, which will raise the sensitivity label of a request to the level of a back-end application when required to access data requested and permitted by the system.

In the preferred embodiment, the authentication module interaction is protected using SSL to encrypt all session traffic. Once the user has been authenticated, the authentication module provides a secret marker (cookie) at step 124 that will be inserted by the user's browser in subsequent communication with this system. The authentication module will also notify the UDE of the marker and the associated sensitivity label at step 128. After a user has been authenticated, the UDE thereafter redirects the user's incoming requests to the appropriate restricted web server. The UDE will enforce a timeout on the marker. If the marker has not been seen by the UDE for a period of time greater then the timeout period, the marker will no longer be valid. To reinstate the connection, a user will have to use the authentication module at step 122 again.

Figure 15 depicts traffic first flowing through the default web server and to the authentication module, and thereafter being redirected by the UDE to a restricted extranet.

The ability to partition a network server is a key component of the trusted operating system of the present invention in providing the level of assurance needed to support critical network and transaction servers. Referring to Figure 12, processes, files, and other resources that have the same sensitivity label are said to be in the same compartment or partition 11. Programs, data, and network interfaces can be split into separate, isolated partitions with restricted access between them. For example back end applications related to specific functions may be stored in

resource controlled or prior to the execution of any process requested.

In the preferred embodiment, the process table is an encrypted file stored on the trusted server that is a read only data structure. The management and control of this table is preferably accessed in the maintenance mode of the system. The management functions may be stored in a separate partition of the storage of the trusted server such that authorized users may execute the maintenance mode while in the administrative mode.

Authorization is an attribute of a user account under the trusted operating system that enables the user to execute a subset of operating system applications and utilities. Privileges are attributes assigned to applications that give them different grades of access to processes or resources. Together, these two sets of attributes enable programs to behave differently for different users. Applications can be extended to define and check new authorizations, providing a standardized protocol by which applications and the operating platform can communicate to grant permissions to users. Authorizations can additionally be used to divide administrative duties into separate roles, which can then be assigned to different users.

The user of the system may execute a process from a particular role which then assigns a sensitivity level of authority to the processes that follow the initial request submitted or received by the processor as generated from an

only receive the role of the lowest level process. Even if a clever user or hacker could embed commands into other process steps, the system would not permit transactions to occur beyond the process level assigned since the primary administrative functions of the system are not enabled or even available to be executed from the storage partition of the system in the secure mode of operation. The printer processing step would therefore be indicated as such in the role assumed such that no other processes may be initiated beyond that role.

In an example where an Internet-based web user requests access to a resource behind the firewall, the system would initially determine from the level originally assigned how to direct the request and what processes it is permitted to initiate. The user may be prompted that the information requested is unavailable if their level is not authorized. The user may be informed that the data requested is not found, or the user may be able to login to the system using an additional security access process.

Network layer encryption is an optional component of this architecture. Because SSL (supported by Netscape, Microsoft, and others) is used to encrypt only HTTP session traffic, the primary purpose of the network layer encryption component is to authenticate an IP address. Host authentication is critical if access to certain functionality (e.g., the administration tools or restricted extranet web servers) is to be allowed based on which host the user is coming from. Although this system permits the

supplied with a customized or personalized version. Some users can connect with one set of privileges while others may execute the same application with differing privileges. For example, a word processor application stored on the server of the present invention may be used by individuals to permit different levels of access to resources or content based on their identities and roles without defining or placing accessibility restrictions on the program with regular permission bits. This would dynamically allow the system to modify the content available to the user where this functionality is not typically provided to a user through the permissions of the word processing program. This enables this system to apply one set of security controls across all of the applications on the system instead of configuring each application separately using proprietary control methods specific to each application software program. By establishing this type of structure at the operating system level, the system administrator does not need to become familiar with the authorization mechanisms of each of the applications but may instead secure the data or process by the global controls of the present invention. In this manner a one-time application of attributes and authorizations may externally determine how the software executes. In a similar manner, instead of authorizing processes within the architecture of the application program, the actual process calls that are executed may be individually restrictively controlled at the operating system by performing this one time attribute definition. In the preferred embodiment, the system levels may be classified or defined according to departmental management levels of authority. In other embodiments the levels may be generated and applied according to a workflow-like methodology where a schedule may be applied to change

This invention allows multiple web servers to be running, each in its own isolated partition. Access to each of the extranet's services is restricted by the UDE 4. In addition, as with all virtual web sites of this invention, individual files can be protected as read-only, thus preventing any malicious modifications to the web site should an exploitable hole be found in a commercial web server software suite installed on the server of the present invention.

Read-only files and directories can be shared by isolated web servers. This makes it possible to use a single copy of common web pages and applications, eliminating duplicate files and reducing administration overhead. The UDE and the extranet web servers can reside on the same physical machine, or the extranets can run on separate systems connected using network-level encryption and the security features of the ASN software.

This invention provides for both local and remote administration. The trusted administration utilities are accessible only to a user who has an account granting access to it, and only when that user is coming from an authorized host. The UDE uses the label applied by the ASN module to determine if an incoming packet can be routed to the trusted administration web server partition 6. In the preferred embodiment, the ASN module restricts administrative functions to those network devices within the intranet. For example, in a Unix-based system, the administrative functions associated with user administration, system

actual partitioning of the system. The system may in this manner be tested prior to implementation. Access to the system and partitioning may also be implemented such that geographic regions of users may be supported. Performance tuning may also be accomplished at this point by integrating simulation software into the system such that any processing routines may be replicated as required to achieve a desired process throughput. During actual use, this type of graphic utility may also assist in providing a representation of a user's process interactions and requests where the output may be directed to administrators using an email process for any processes that deviates from the predefined paths defined in the original system. The graphic representation that is ultimately chosen and tested may be frozen such that no other processes may be permitted. This graphic model of the system is modified into a tabular or linked list form that may then be used by the UDE and the security gate 8 to control process activities. In the same manner that each individual process may be analyzed, an overall system level analysis may be performed where each process may be traced back to the originating process stream or to determine overall process performance constraints.

As a reliable, secure basis for Internet transactions of all types, the present invention is a top-shelf security solution for any type of database gateway system. For example, medical organizations wishing to provide patient histories or prescription records to local and remote users via the Internet are necessarily concerned with patient confidentiality. This system provides a security framework for authorizing access for Internet

authorized for one partition do not gain access of any type to services in other partitions.

In a medical environment, patient histories can be securely stored and delivered over the Internet between hospital branches, while separate web services provide prescription information to authorized pharmacists. Unauthorized users will be denied access altogether or routed to the organization's public web server. The labeling mechanism ensures that pharmacists authorized to use the prescription information partition are prevented from accessing information or programs in the patient history partition.

This invention can be used to create web sites that are accessible and readable across public networks, but remain completely protected from unauthorized outside modification. Using this invention, web pages can be stored in partitions that allow read access but not write access. In this way, administrators are assured that malicious parties cannot vandalize the web site by changing existing information, posting obscene materials, and otherwise altering the web site content. Web sites that incorporate this system are also protected from attack through the exploitation of bugs in web server applications.

The types of attributes that may be assigned are only limited by the available storage on the computer and to the extent that they provide some form of value or control within the system. A level of practicality may be employed by the administrators of the system to limit the levels

operating systems. The information on these transactions is appended to an audit trail in an isolated partition, which is protected by both discretionary and mandatory access control mechanisms. This approach prevents intruders from covering their tracks and eliminating traces of penetration attempts. By protecting the audit record, sufficient evidence is maintained to support litigation and law enforcement actions.

5. The method of claim 1 wherein the incoming request comprises indicia identifying the source of the request.

6. The method of claim 5 wherein the indicia is representative of the users IP address.

7. The method of claim 1 further comprising the step of decrypting the request.

8. A method for assigning control and access attributes to data objects for a commercial software product executing on a trusted server, where access to the commercial software product is restrictively managed by the trusted server, the trusted server performing the steps of:

a. executing the commercial software product in a configuration mode;

b. determining at least one process to be accessed by the commercial software product while in the configuration mode;

c. receiving administrator input for a least one sensitivity level to be assigned to the data files and processes;

d. determining extended attributes to be applied to the processes and data files of the commercial software product;

e. applying the determined extended attributes for the received administrator sensitivity level to the processes and data components of the commercial software product; and

11. The method of claim 9 wherein the process is limited to those defined by the processes configured during the configuration mode.

12. The method of claim 9 wherein the user indicia comprises authorization information used in conjunction with the assigned sensitivity level to determine a role of the user.

13. The method of claim 9 further comprising the step of decrypting the request related to the commercial software product.

14. The method of controlling access to the processes of a trusted server that performs a secondary access check in addition to the owner, world and group access with associated read, write and delete authorization, comprising the steps of:

a) segmenting the control aspects of the administrative user accounts of the trusted server such that all administrative functions cannot be performed by one administrative account;

b) extending the attributes of the file system to include at least a sensitivity level attribute;

c) assigning the sensitivity level attribute to each of the files and processes stored in at least one storage location in the file system;

d) generating a table of processes of the system where the table contains relationships to the roles permitted to execute the processes;

b) processor means for receiving requests and executing processes in response to the user requests;

c) assignment means for assigning a sensitivity level associated with the request for a data object;

d) upgrade downgrade enforcer means for interpreting the request to determine a first destination to direct processor to retrieve the process and extended attributes, the upgrade downgrade enforcer comparing the attributes to the assigned sensitivity level, and passing the favorably compared processes to the processor for execution;

e) the processor identifying processes that require access to secured storage partitions and directing these processes with the sensitivity level to a security gate means;

f) the security gate means for receiving the sensitivity level and interpreting processes requiring data from partitioned storage, the security gate means retrieves the requested data if the data sensitivity level is favorably compared to the received sensitivity level.

17. The trusted server of claim 15 wherein the request is received from a networked user on a user computer.

18. The trusted server of claim 15 wherein the request is generated by the processor of the trusted server as an output of a prior request.

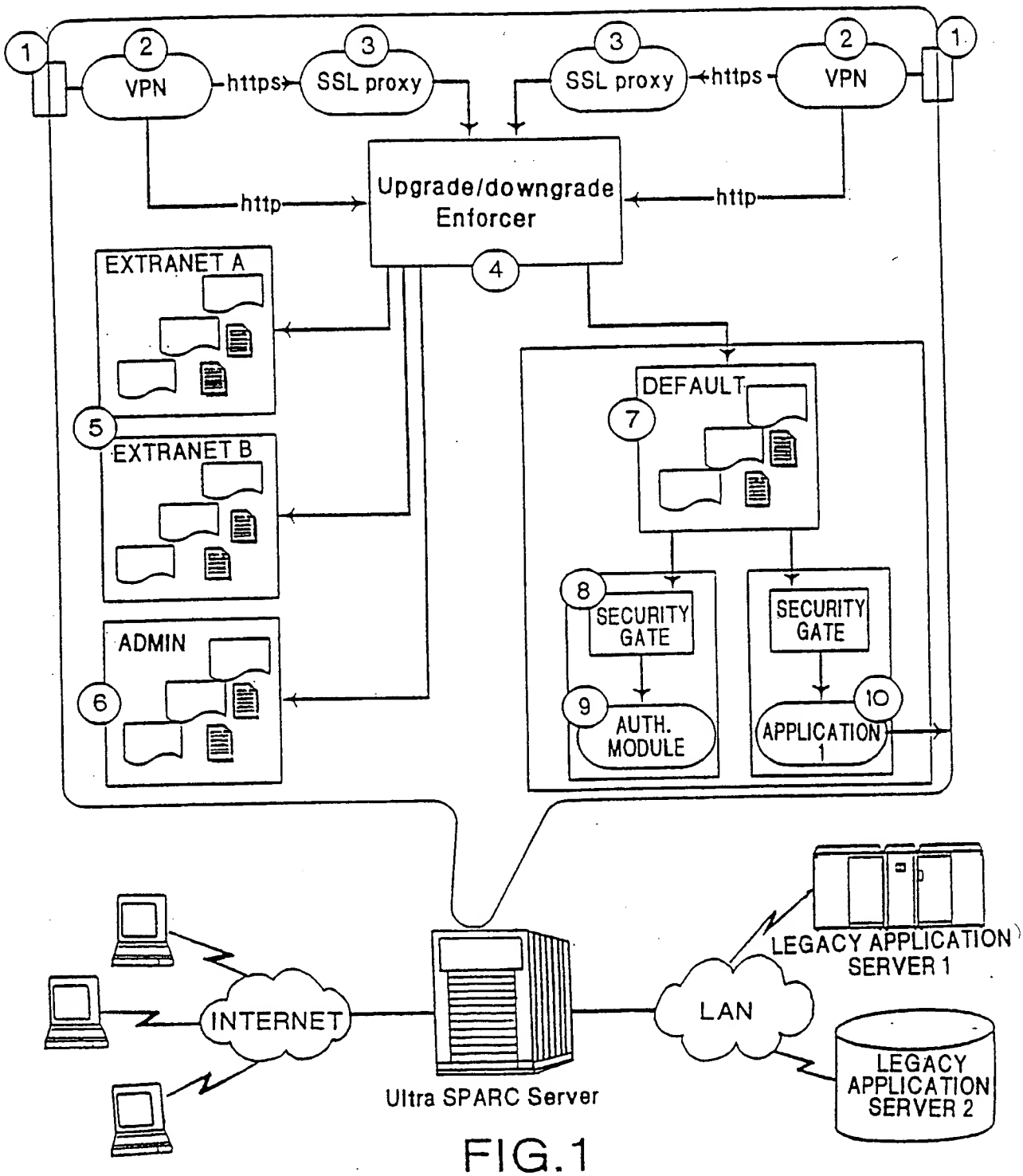

1
2
VPN
https
3
SSL proxy
3
SSL proxy
https
2
VPN
1
http
Upgrade/downgrade
Enforcer
http
4
EXTRANET A
5
EXTRANET B
ADMIN
6
DEFAULT
7
8
SECURITY GATE
9
AUTH. MODULE
SECURITY GATE
10
APPLICATION 1
INTERNET
Ultra SPARC Server
LAN
LEGACY APPLICATION SERVER 1
LEGACY APPLICATION SERVER 2

FIG.1

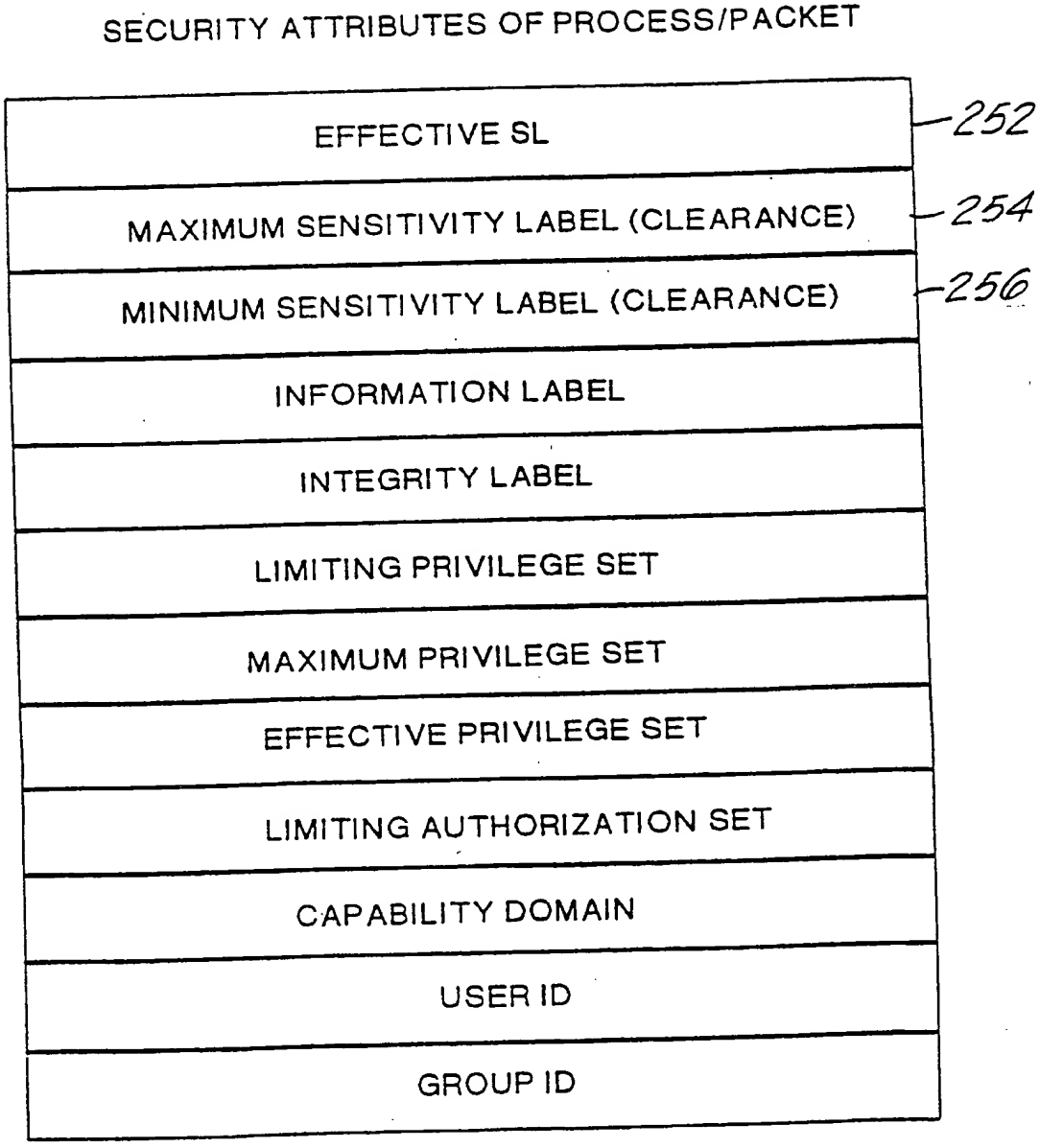

SECURITY ATTRIBUTES OF PROCESS/PACKET
EFFECTIVE SL
252
MAXIMUM SENSITIVITY LABEL (CLEARANCE)
254
MINIMUM SENSITIVITY LABEL (CLEARANCE)
256
INFORMATION LABEL
INTEGRITY LABEL
LIMITING PRIVILEGE SET
MAXIMUM PRIVILEGE SET
EFFECTIVE PRIVILEGE SET
LIMITING AUTHORIZATION SET
CAPABILITY DOMAIN
USER ID
GROUP ID

FIG.3

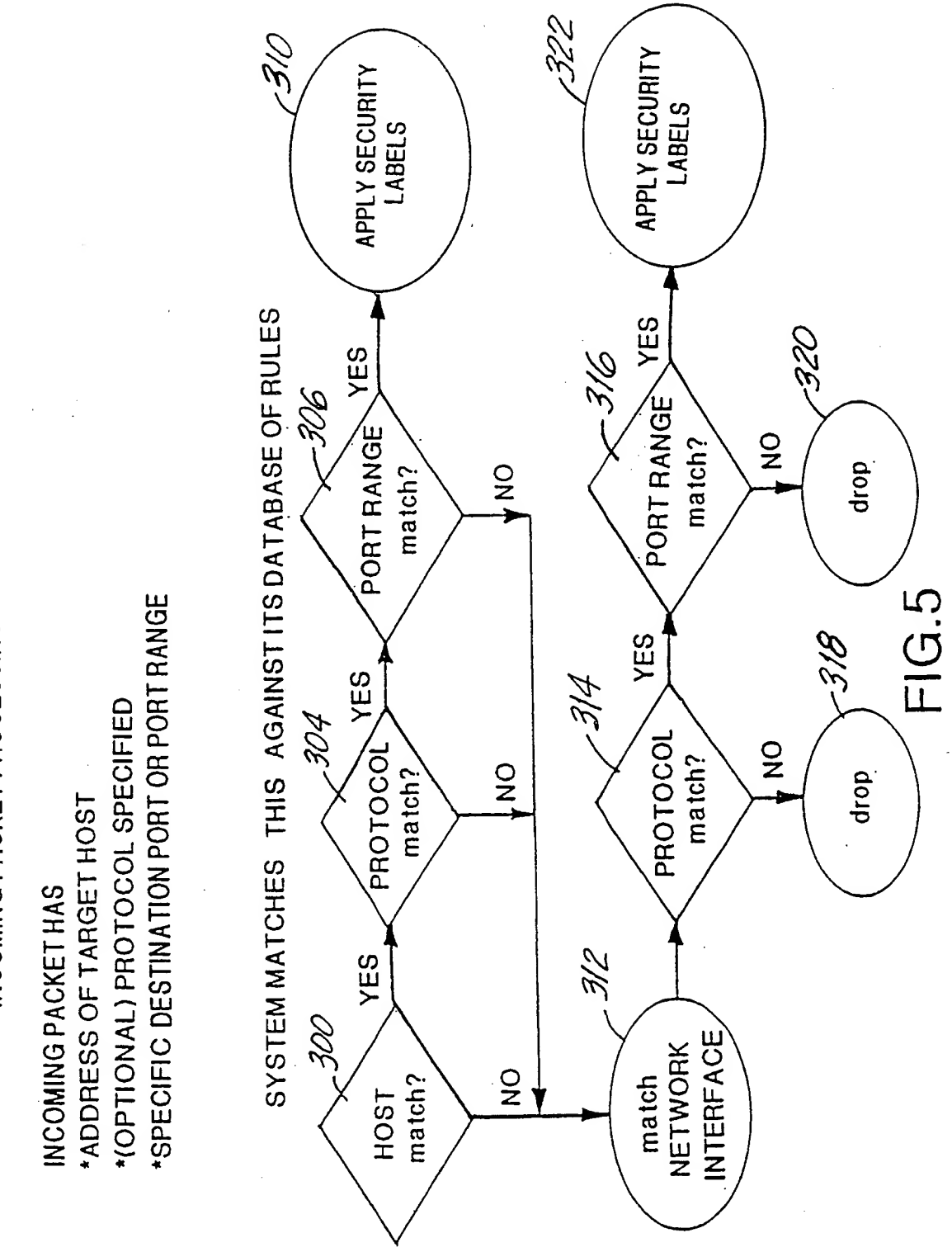

INCOMING PACKET PROCESSING
INCOMING PACKET HAS
*ADDRESS OF TARGET HOST
*(OPTIONAL) PROTOCOL SPECIFIED
*SPECIFIC DESTINATION PORT OR PORT RANGE
SYSTEM MATCHES THIS AGAINST ITS DATABASE OF RULES
HOST match?
300
YES
NO
PROTOCOL match?
304
YES
NO
PORT RANGE match?
306
YES
NO
APPLY SECURITY LABELS
310
match NETWORK INTERFACE
312
PROTOCOL match?
314
YES
NO
drop
318
PORT RANGE match?
316
YES
NO
drop
320
APPLY SECURITY LABELS
322

FIG.5

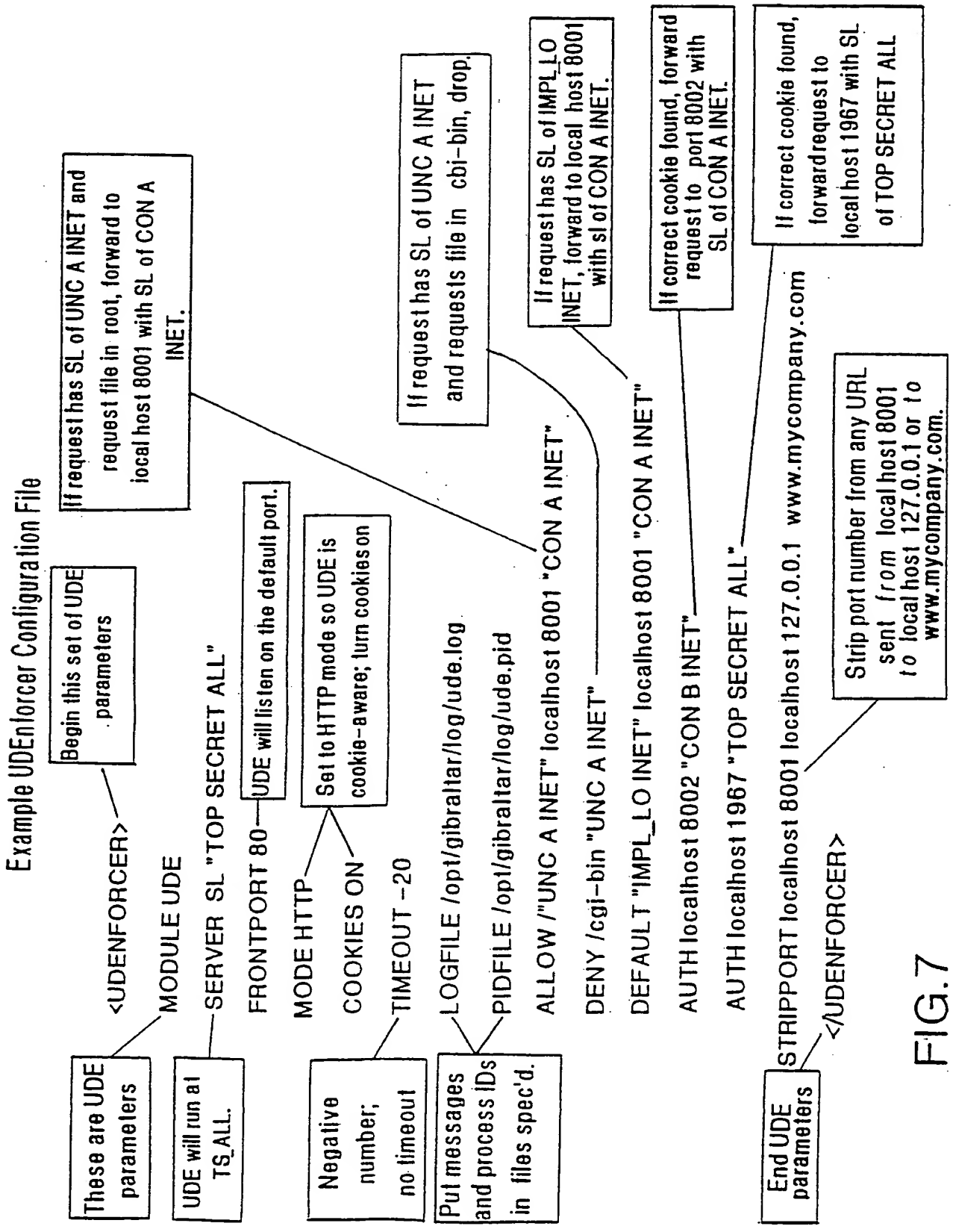



FIG.7

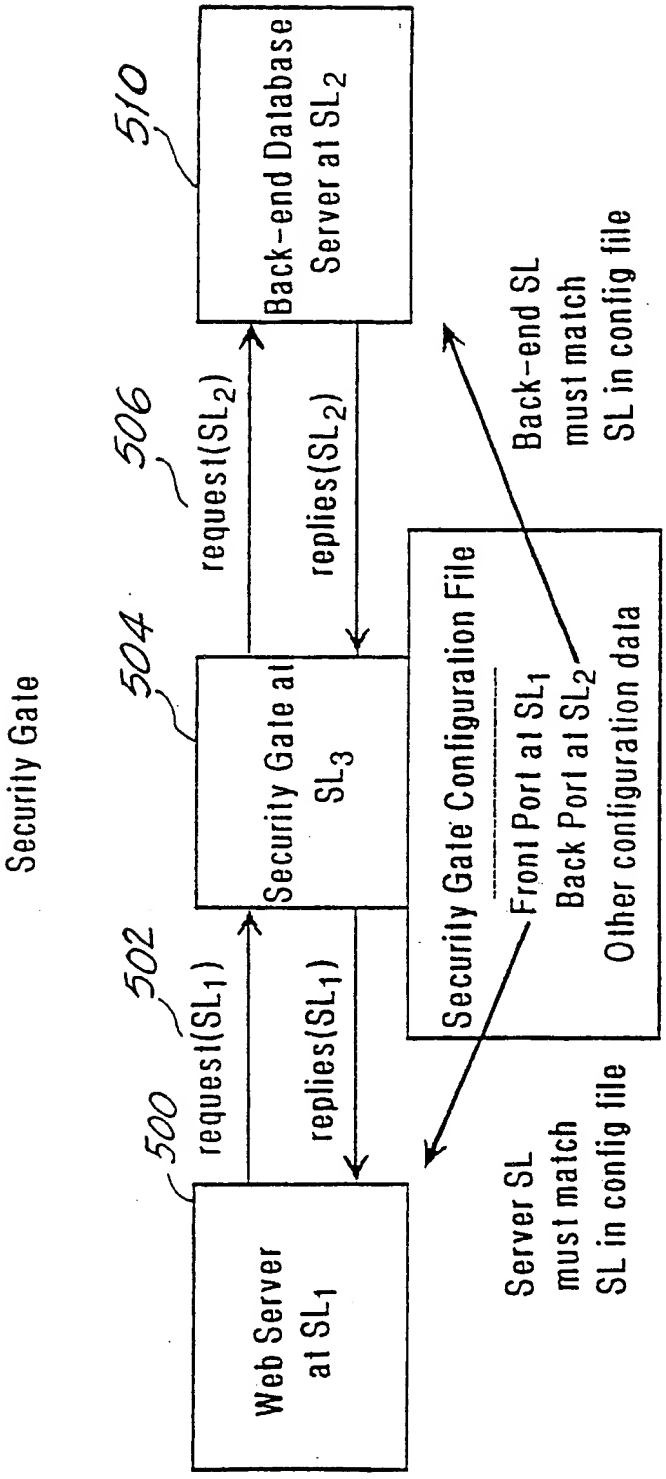

Securily Gate
500
Web Server at $SL_1$
502
request($SL_1$)
replies($SL_1$)
504
Securily Gate at $SL_3$
506
request($SL_2$)
replies($SL_2$)
510
Back-end Database Server at $SL_2$
Securily Gate Configuration File
Front Port at $SL_1$
Back Port at $SL_2$
Other configuration data
Server SL must match SL in config file
Back-end SL must match SL in config file
Security Gate (at $SL_3$) has priority to operate at $SL_1$, $SL_2$, and/or $SL_3$.

FIG.9

EXECUTION OF BINARY

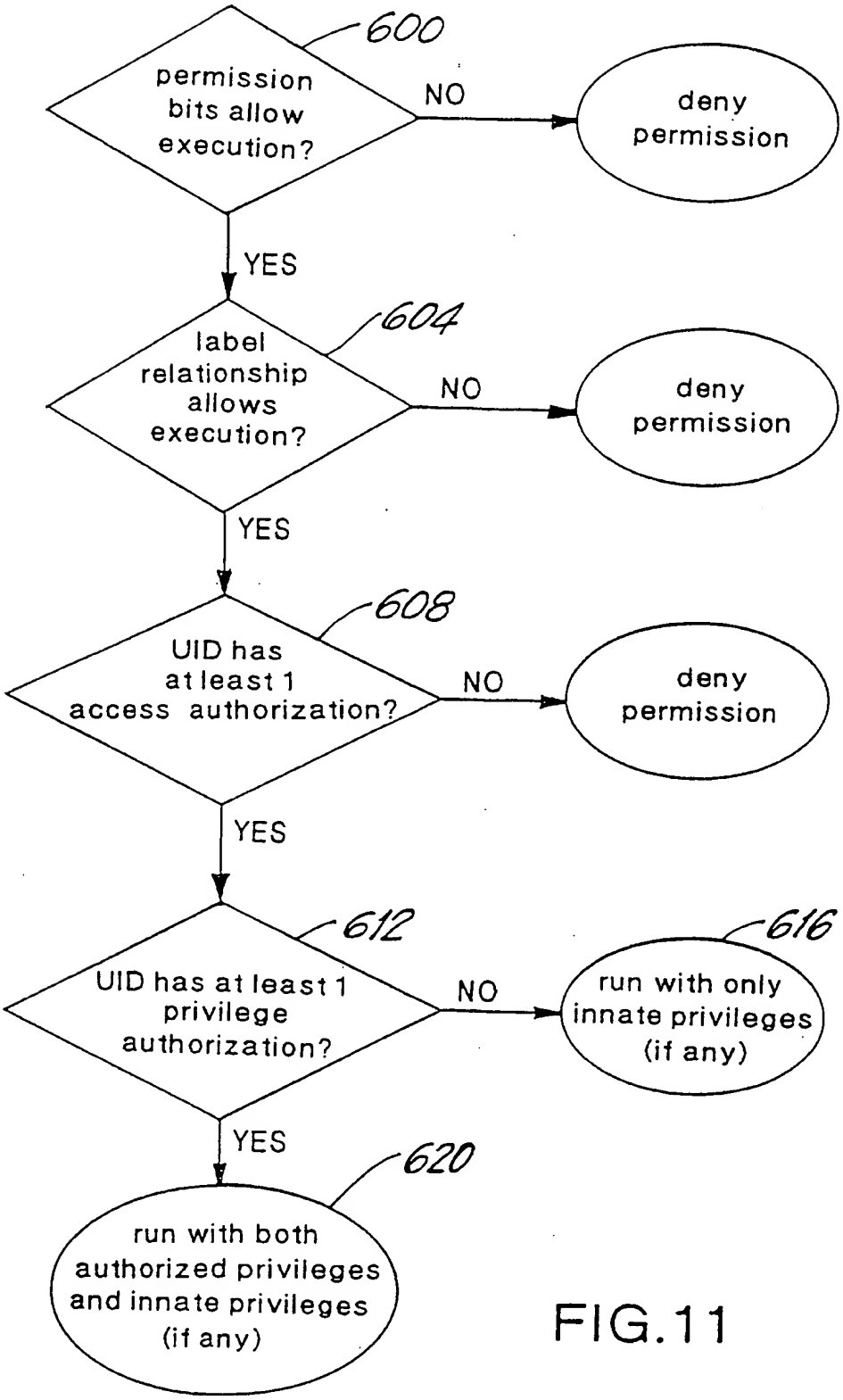



FIG.11

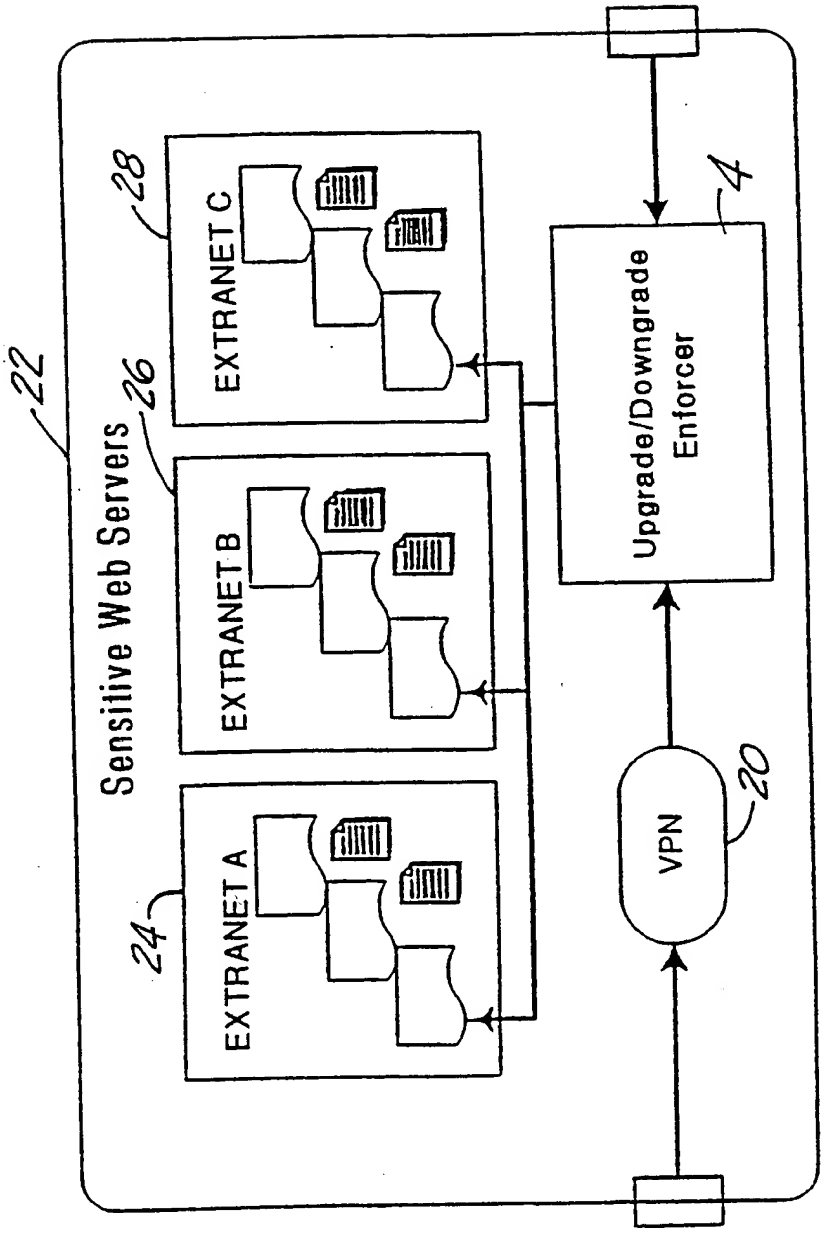

22
Sensitive Web Servers
28
EXTRANET C
26
EXTRANET B
24
EXTRANET A
4
Upgrade/Downgrade Enforcer
20
VPN

FIG.13

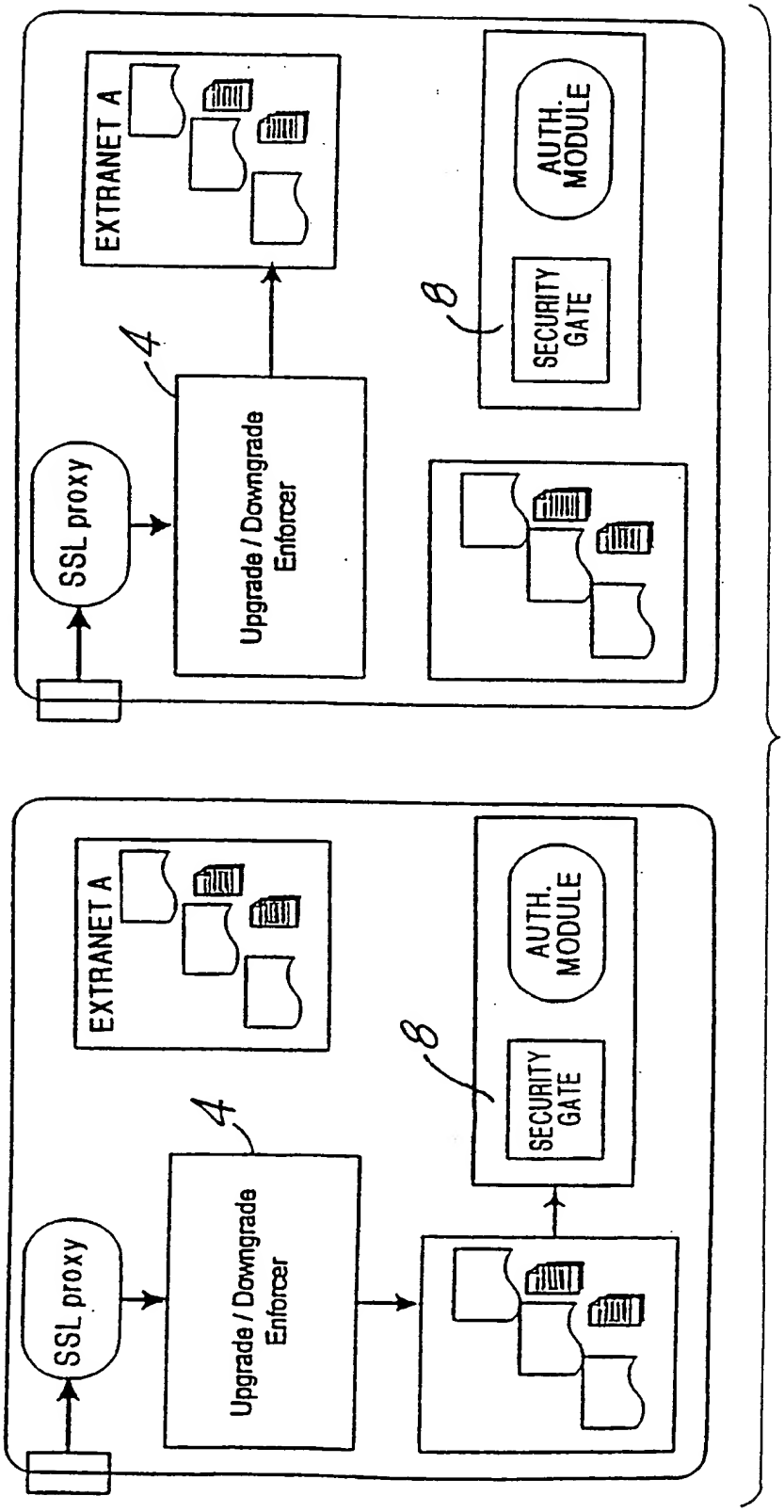



FIG.15

# INTERNATIONAL SEARCH REPORT

International application No.
PCT/US99/22331

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :G06F 13/00
US CL :713/201

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 713/201, 200; 709/225, 229

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EAST
search terms: sensitivity levels, extended attributes

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
|---|---|---|
| Y, P | US 5,903,732 A (REED ET AL.) 11 May 1999, col. 3, line 41 - col. 8, line 4. | 1-18 |
| Y, P | US 5,845,068 A (WINIGER) 01 December 1998, col. 3, line 31 - col. 9, line 11. | 1-18 |
| X | US 5,596,718 A (BOEBERT ET AL.) 21 January 1997, col. 3, line 66 - col. 8, line 5, and col. 9, line 41 - col. 10, line 6. | 19 |
| Y | STEEN, W. et al. "NetWare Security" 31 December 1996, New Riders Publishing, Chpt 10: pp 248-277. | 1-18 |

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

| | | | |
|---|---|---|---|
| * | Special categories of cited documents: | "T" | later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention |
| "A" | document defining the general state of the art which is not considered to be of particular relevance | "X" | document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone |
| "E" | earlier document published on or after the international filing date | "Y" | document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art |
| "L" | document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) | "&" | document member of the same patent family |
| "O" | document referring to an oral disclosure, use, exhibition or other means | | |
| "P" | document published prior to the international filing date but later than the priority date claimed | | |

| | |
|---|---|
| Date of the actual completion of the international search<br>10 JANUARY 2000 | Date of mailing of the international search report<br>07 FEB 2000 |
| Name and mailing address of the ISA/US<br>Commissioner of Patents and Trademarks<br>Box PCT<br>Washington, D.C. 20231<br>Facsimile No. (703) 305-3230 | Authorized officer<br>STEPHEN ELMORE Joni Hill<br>Telephone No. (703) 305-3800 |

Form PCT/ISA/210 (second sheet)(July 1992)★